***App-etizers at SC***

***Test Cases***

**Version 1.0**

**Prepared by:**

**Yuming Fei, Renhe Song, Xizhe Ma, Yuting Gao, Peng Gan, Marika Perlmutter**

**CSCI 201: Principles of Software Development**

**Search Page**

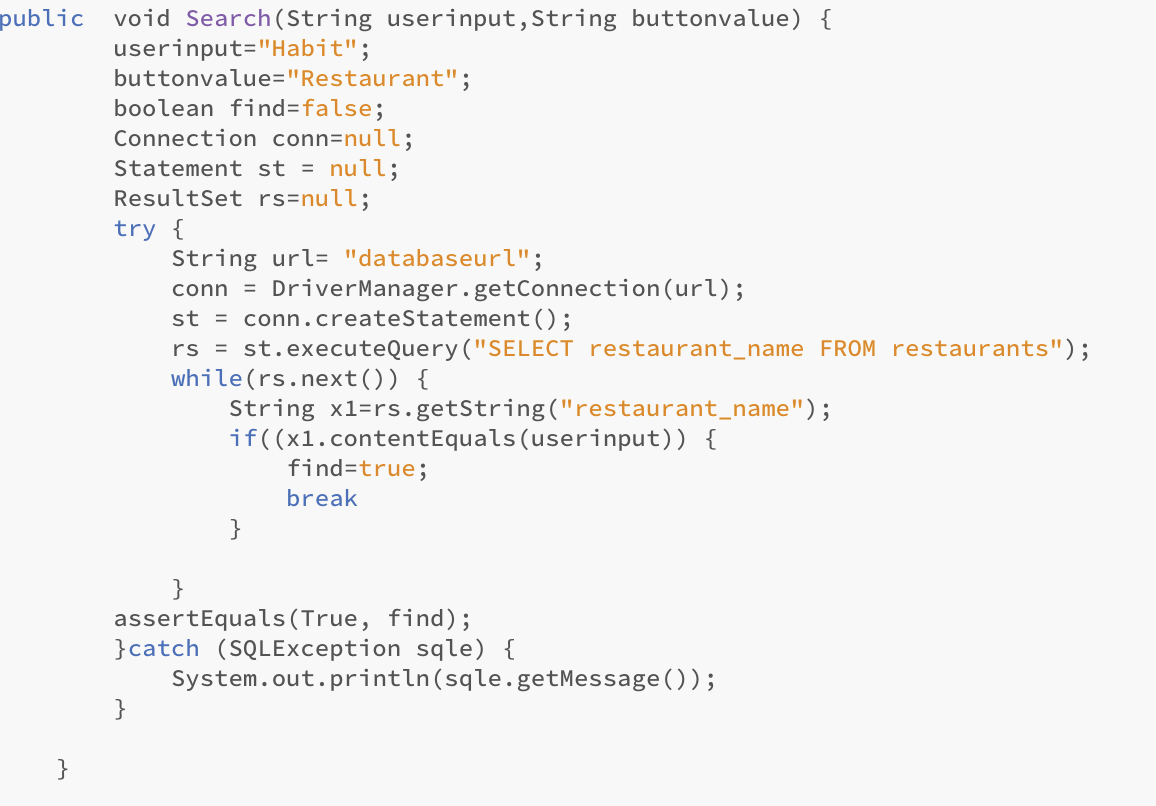
* Test Case 1

White Box Test - test whether the search result matches user input and the keyword value. Search “hamburger” and choose the button “keyword”. The web page should direct to Dish page and display hamburgers in different restaurants.



* Test Case 2

White Box Test - test whether the search result matches user input and the keyword value. Search “Habit” and choose the button “Restaurant” . The web page should direct to Restaurant page and display the restaurant Habit.



* Test Case 3

White Box Test - test whether the page can display “sign out” and “profile” buttons after the user logs in. test whether the page display “sign up” and “log in” button when user is not logged in. Signing in/out the page and without reloading to see if the page can display the correct buttons. 

* Test Case 4

White Box Test - test whether sign up, log in, and contact buttons work. Clicking the sign up button should pop up the window for registration. Clicking the login button should pop up the window for logging in.

* Test Case 5

Unit Test - select SQL code in the search method in the Search servlet. Test it to see if it can return correspond information from the Restaurant table and Dish table. The web console should print out information of relevant restaurants and dishes.

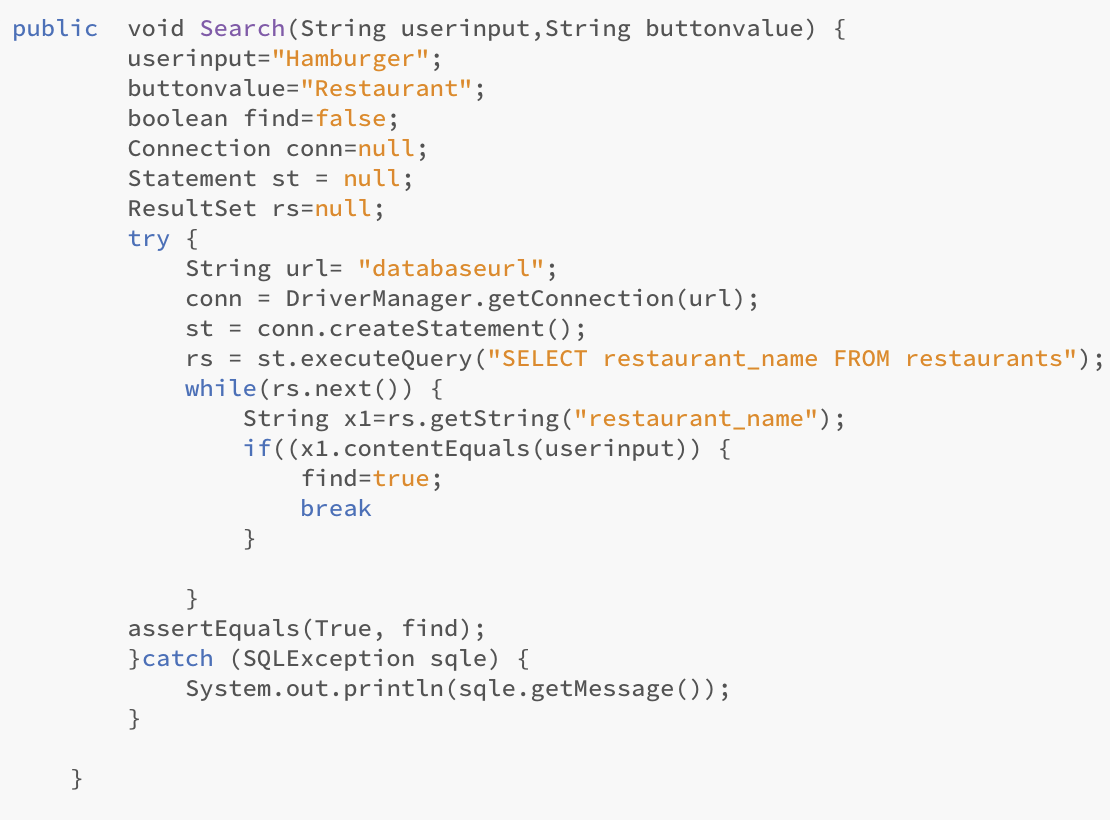
* Test Case 6

Stress Test - type random words or characters in the input text field and select the keyword button. Test what result will be given. The web page should direct to the Dish page but display one error message “No Result is Found!”



* Test Case 7

Stress Test - test what will happen if user input is irrelevant to keyword value. Search “hamburger” and choose the button “Restaurant”. The web page should direct to the Restaurant page and display the Hamburger restaurant. Otherwise, it should display “No Result Found!”



**Video Section**

* Test Case 1

White Box Test - test whether the web page can display video appropriately by choosing different browsers such as Chrome, Safari, and IE. User should be able to watch the website video through these web browsers.

* Test Case 2

White Box Test - test whether the web page can display the slide show and change it automatically by choosing different browsers such as Chrome, Safari, and IE. User should be able to watch the slide shows through these web browsers.

* Test Case 3

Unit Test - adding some random characters in video tag src attribute to see if an appropriate error message can be displayed. The web page should not display a video, but instead display an error message “No Video Source is Found!”

* Test Case 4

Unit Test - adding some random characters in image tag src attribute to see if an appropriate error message can be displayed.The web page should not display an image, but instead display an error message “No Image Source is Found!”

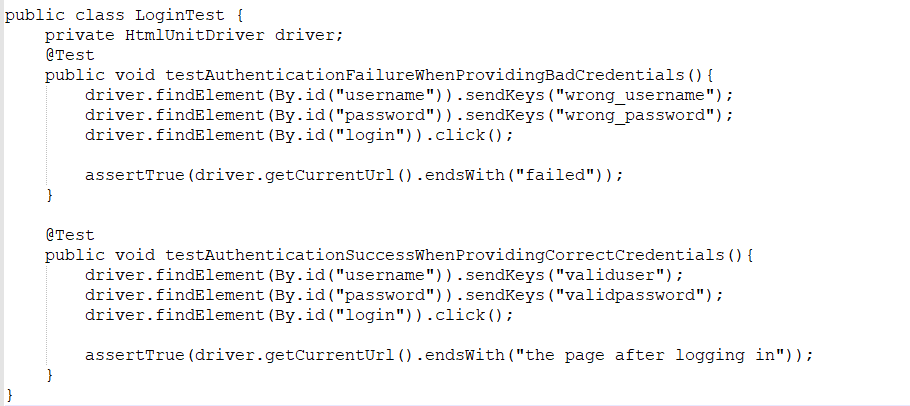
* Test Case 5

Unit Test - click the PushSlideButton in the function ShowSlides(index) to see if it can push the slide forward and backward. The slide show should move to the next image.

**Login and Register Section**

- TestCase1  
 White Box Test – test the login functionality by specifying a username that exists and a password that does not match. The user should be taken back to the login page with a message “Invalid password or username.”

* Test Case2  
   White Box Test – test the login functionality by specifying a username that does not exist. The user should be taken back to the login page with a message “Invalid password or username.”
* Test Case3  
   White Box Test – test the login functionality by specifying a username that exists and a password that matches. The user logged in successfully and the signs on the right corner will indicate that.
* Test code for the login functionality



**Database Schema**

* Test Case1

Black Box Test - Verify database has been constructed correctly and tables are identical to the Database Schema. By Login to the Gcloud Sql and manually verify using “show create table”. The parameter and types are identical to the Database Schema.

- Test Case2

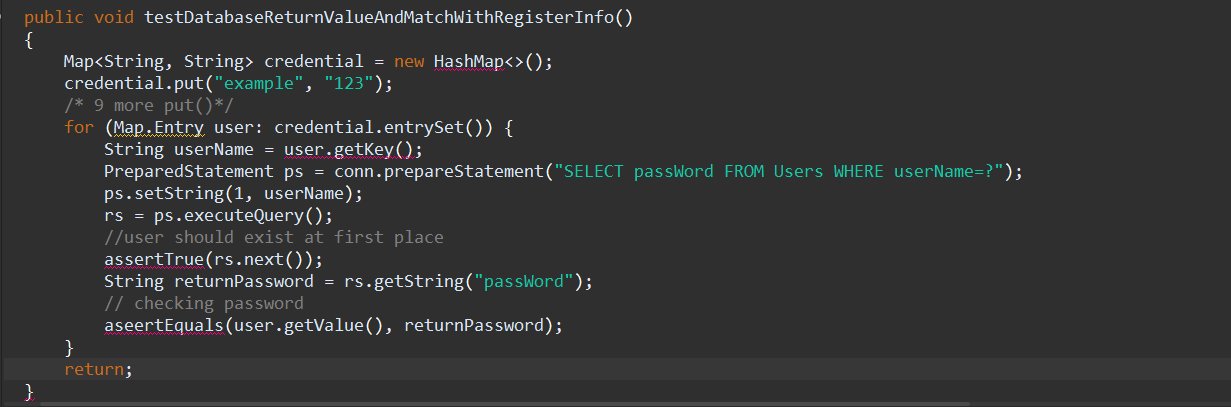
Black Box Test - Verify that upon inserting to tables, data are showing correctly on the database. Login to the Gcloud Sql and Insert one test entry to each table and check that data is displayed correctly. There should be one entry for each table.

- TestCase3

Black box test - Verify entry is modified whenever admin publish an update to the database. White box test - Modify all tables by using “alter table” and check that data reflect correctly. Data reflect what we have modified through the command, foreign key that doesn’t exist should report error in our system logging.

* TestCase4  
  Black box test - When users register on the Register Page, database should correctly reflect users information. Manually register 10 test users on webpage, log in to Gcloud and verify that the account name and password on the database are identical to the information that was entered on the webpage.
* TestCase 5

White box test - Database will return the credential to the server when user login in order to perform any verification routine. Verify this by log in the 10 test users on the login page in the TestCase4. Upon success, database should return correct user information to the server so that the server can do necessary verification process. This could be done either printing the return information or creating a test function.



* TestCase 6

Black box test - Users should be able to log in to the database successfully after the first try. This means that data on the “users” table is constant and will not fail because of how many times users have logged in. To test this, manually log in and log off the same users for 10 times, and verify that database have the exact information stored.

* TestCase 7

Black box test -Database should reflect what users have bookmarked the favorite dishes. Verify this by login a test users and pick 5 random dishes from the dish page and bookmark the dishes. Login to Gcloud mysql and check the table “users\_favourite” has the corresponding dishes.

- TestCase9

Black box test - “Users\_favorite” Table should not have any favorite dishes that users have removed. Verify this by removing two of the favourite dishes given in TestCase 8. Login to the database and check that any rows have been removed correctly.

* TestCase 10

Stress Test - The most phenomenal structure in database is the review section. Some review might reach 500+ words and this essentially put a lot of stress on the database in terms of time performance. To solve this, some order-by optimization are needed(create extra index). The test will make sure the result produced by the optimization is accurate.

**Restaurant Page**

* White-Box Testing (see code from Search Page section):
  + Test 1: Verify searching functionality from main Search Page applies to search bar in header of Restaurant Page.
* Black-Box Testing:
  + Test 1: Page loads in a reasonable span of time with all required page components (site logo, keyword search bar, keyword/restaurant selector, search button, username icon, username link, logout link, search result information header, filters dropdown, individual restaurant details including number and name, logo, star rating, address, price indicator, bookmark button, and bookmark indicator).
  + Test 2: Restaurant results displayed are visually recognizable as relevant to restaurant name or keyword searches.
* Unit Testing:
  + Test 1: Clicking on site logo redirects back to site main page.
  + Test 2: Clicking on search button refreshes page with current keyword/restaurant results.
  + Test 3: Clicking on username icon redirects to user detail page.
  + Test 4: Clicking on username redirects to user detail page.
  + Test 5: Clicking on “Log out” text logs out user.
  + Test 5: Clicking on restaurant logo redirects to restaurant detail page.
  + Test 6: Clicking “Add to bookmark” changes bookmark icon shading and adds restaurant to bookmark database.
* Input Testing:
  + Test 1: Entering a keyword into the search bar returns a list of restaurants associated with that keyword.
  + Test 2: Entering a keyword with no valid results displays “No restaurants matched your search!”
  + Test 3: Entering a null (“”) keyword displays “No restaurants matched your search!”
  + Test 4: Entering a restaurant into the search bar returns a restaurant or list of restaurants matching that name.
  + Test 5 Entering a restaurant with no valid results displays “No restaurants matched your search!”
  + Test 6: Entering a null (“”) restaurant name displays “No restaurants matched your search!”
* Stress Testing:
  + Test 1: Extremely long text inputted into the search bar is truncated according to character limit.
  + Test 2: Capacity of bookmark page sufficient to hold all added restaurants, if applicable.
  + Test 3: Extremely long list of restaurant results display on page and are not constrained by web page length.
* Regression Testing:
  + Test 1: Rating restaurants on Rating Page results in cumulative star rating output on Restaurant Page.
  + Test 2: Bookmarking restaurants on Restaurant Page results in addition to user’s Bookmark database, and display on user information page.

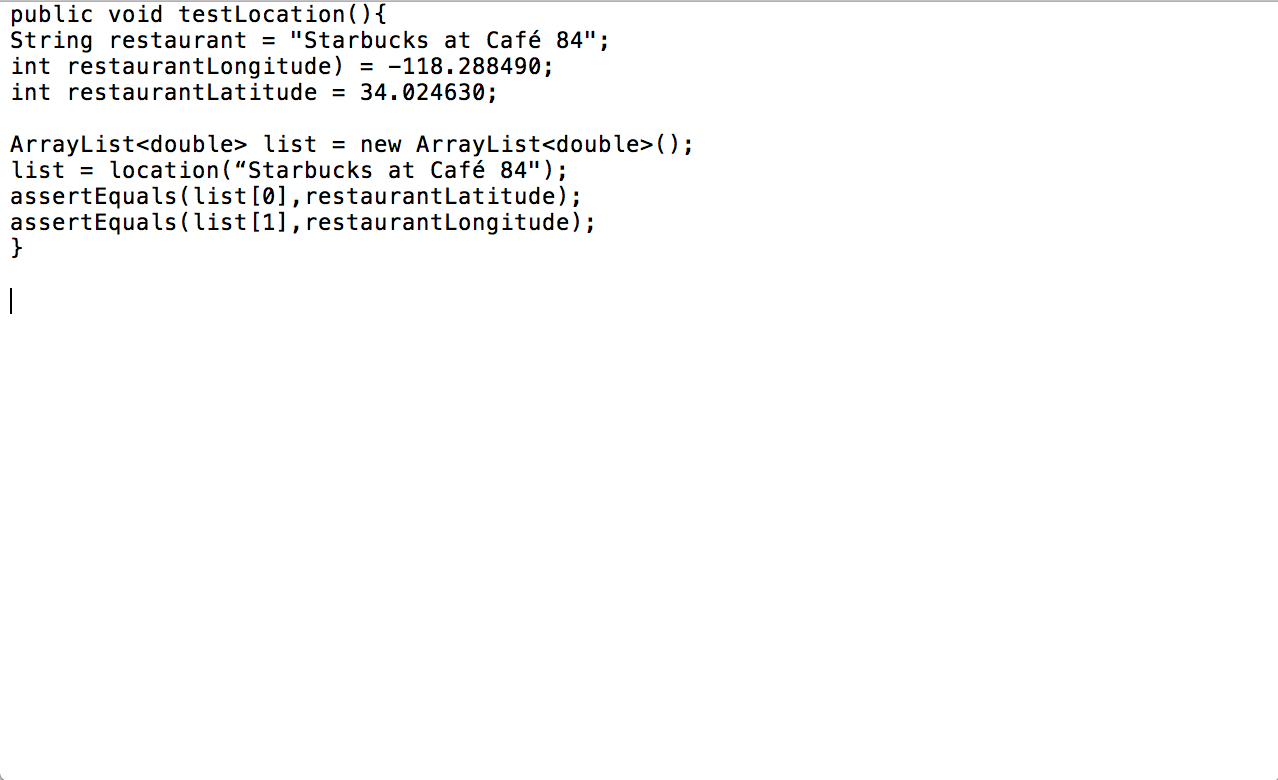
**Restaurant Detail Page**

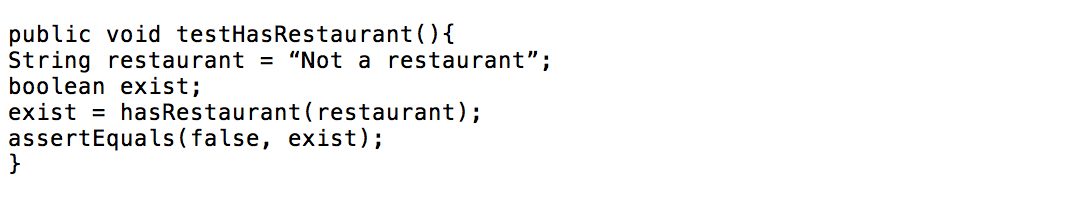
* White-Box Testing (see code from Search Page section):
  + Test 1: Verify searching functionality from main Search Page applies to search bar in header of Restaurant Page.
* Black-Box Testing:
  + Test 1: Page loads in a reasonable span of time with all required page components (site logo, keyword search bar, keyword/restaurant selector, search button, username icon, username link, logout link, search result information header, individual restaurant details including name, logo, pictures of food, menu, star rating, address, and menu item price indicator).
  + Test 2: Restaurant result displayed matches the restaurant selected on the Restaurant Page by the user.
* Unit Testing:
  + Test 1: Clicking on site logo redirects back to site main page.
  + Test 2: Clicking on search button refreshes page with current keyword/restaurant results.
  + Test 3: Clicking on username icon redirects to user detail page.
  + Test 4: Clicking on username redirects to user detail page.
  + Test 5: Clicking on “Log out” text logs out user.
  + Test 5: Clicking on restaurant logo redirects to restaurant detail page.
  + Test 6: Clicking on images of food at page left redirects to associated review pages for those dishes.
* Stress Testing:
  + Test 1: Extremely long text inputted into the search bar is truncated according to character limit.
  + Test 2: Extremely long list of menu items display on page and are not constrained by web page length.

**Dish Page**

* Test 1: White-box if there is nothing entered in the search bar, and the user hit search button, the page should be refreshed and stays the same as the search page
* Test 2: Whitebox - if we click on a dish and save it as favorite, the favorite sign should be filled with color instantly, on the other hand, if we unfavorite it, it should turn to unfilled sign instantly without refreshing the page
* Test 3: Blackbox - search the dishes on the web page, and the relevant searches appear, for example typing in burger, different burger from different restaurants should appear
* Test 4: Blackbox - clicking the logo on the top left corner should redirect the page to the main page where the video plays
* Test 5: Edge case testing: If we enter a restaurant name under the dish page search bar, the result should not show any dishes since the user entered the restaurant name, and not the dish related ingredient
* Test 6: Stress testing, enter a very long list of ingredients in the search bar
* Test 7: Black box - if a dish is clicked, the user should be redirected to the specific description of the dish, about the price, the ingredient, and the restaurant that has it.

**Map Section**

* Test Case 1 test the functionality of Location which returns the Longitude and latitude of specific restaurant. Test if the location function would actually return the longitude and the latitude of the given restaurant, in this case the restaurant Starbucks at Café 84 has 34.024630,-118.288490.****
* Test Case 2 test the functionality of hasRestaurant which returns a boolean indicates that if the restaurant is on the map. In this case the restaurant “Not a restaurant” is not on the map.

****

**Comment and Review section**

* TestCase 1  
  White Box Test - test whether the content of the review and the rating stars matches user input. Type “Good food” in the review textbox and select four stars for the food. The user then clicks the submit button and get the corresponding result of the review with stars and comments from the reloaded page.
* TestCase 2  
  White Box Test - test whether the image for the review and the rating stars matches user input. Upload a random picture in the review textbox and select four stars for the food. The user then clicks the submit button and get the corresponding result of the review with starsand image from the reloaded page.
* TestCase 3  
  White Box Test - test whether the content of the review, the content of the review and the rating stars matches user input. Type “Good food” in the review textbox, upload a random picture and select four stars for the food. The user should then click the submit button to see if he or she can get the corresponding result of the review from the reloaded page.
* TestCase 4  
  White Box Test - test the functionality of “like” a comment. The user clicks an existing comment to see if the likes of that comments add correspondingly.
* TestCase 5  
  White Box Test - test the functionality of “unlike” a comment. The user clicks an existing comment he or she clicked “like” to see if the likes of that comments reduce correspondingly.
* TestCase 5  
  Regression Test - test the limit of String for the content of the review. Choose 4 stars and type twice the length of characters that a String can hold and then submit. Test what result will be given. It should only store part of comments the user typed.
* TestCase 5  
  Regression Test - test empty input in the textbox. The user enters nothing in the textbox and does not click on the rating stars. Test what result will be given. It should not be stored in the database.

**User information page**

* TestCase 1  
  Black Box Test - test the functionality of “Overview”. Select “Overview” in the sidebar, the recent activities of the user should appear on the page.
* TestCase 2  
  Black Box Test - test the functionality of “Bookmarks”. Select “Bookmarks” in the sidebar, the bookmarks of the user should appear on the page.
* TestCase 3  
  Black Box Test - test the functionality of “Reviews”. Select “Reviews” in the sidebar, the bookmarks of the user should appear on the page.
* TestCase 4

Black Box Test - recent activity should be listed in chronological order(most recent one appear on the top, and the oldest on the bottom). Check the timestamp of the activity.

- TestCase 5

Black Box Test - when there is no activity for new account, there should not be garbage value display on any of the above three sections(Overview, bookmarks, reviews)

* + TestCase 6

Black Box Test - Activities are recorded error free, i.e, user should not see any information that they haven’t done to the system. Bookmarks should contain all the dishes that users actually added themselves, and no else is allowed.